# Parallel Distributed Computing for Molecular Dynamics: Simulation of Large Heterogeneous Systems on a Systolic Ring of Transputers

*By Helmut Heller and Klaus Schulten, Ph.D.*

For the purpose of molecular dynamics simulations, we have built a parallel computer based on a systolic double ring architecture, with Transputers as computational units, programmed in occam II. The design is very compact and achieves high reliability even on continuous duty cycles of several months. A parity check logic ensures the correctness of data stored in the external RAM. Our program EGO, designed for simulation of very large molecules (up to 50,000 atoms), is compatible with the programs CHARMm and X-PLOR. EGO increases its computational throughput nearly linearly with the number of computational nodes. Results of simulations on the Transputer system of a lipid bilayer are presented.

### Introduction

A principal aim of molecular biology is to understand the relationship between structure and function exhibited by the molecules of life. Static structures of several of these important molecules have been resolved using experimental techniques such as X-ray crystallography and nuclear magnetic resonance. Molecular dynamics simulations can be used to refine these structures and to determine their stability. For some time, it was assumed that the static structure along with knowledge of the chemical bonds and the typical atomic charges involved would be sufficient for a full explanation of a molecule's function. However, in the recent decade, it became increasingly clear that the *motion* of the individual atoms plays a crucial role in determining many aspects of molecular properties and interactions. The details of these processes are often very difficult to measure experimentally, if at all. The information of interest can then only be obtained by computer simulations. Reviews of this exciting field, and examples of many applications, can be found in the literature.[24,3,31,28]

Currently, many groups are developing approaches to allow an increasingly faithful representation of the dynamics of biological macromolecules in computer programs.

One effort is to include in the simulation as much as necessary of the natural environment of the protein under investigation, e.g. membrane and water (see Figure 1, page 12).

In the near future, this will make it possible to leap beyond simple analysis of structure-function relationships and allow significant contributions to biology, ▶

## LAW BRIEFS

protects the text of a software program, but not the "behavior" of that program. The suit involved an action by Computer Associates International Inc. against Altai Inc. for copyright infringement, trade secret misappropriation, and unfair competition. Relying heavily on a technical report written by an independent expert appointed by the Judge in this case, the court held that the "behavior" of a program, defined as the operation of the program as seen from the user's point of view, is now protectable under copyright law.

This decision is expected to have a significant impact on the software industry. One major signficance of this case is the fact that it contradicts the now famous holding of *Whelan v. Jaslow,* in which the U.S. Court of Appeals for the Third Circuit held that legal protection of software under the copyright laws extends beyond the iteral code, or text. The recent Second Circuit decision severely limits the scope of protection granted to software programs under the copyright laws, perhaps shifting the burden of providing adequate protection for software inventions to the patent laws. Computer Associates has not decided whether they will appeal the case to the Supreme Court.

(DISCLAIMER: the material contained herein and in our "CAMD and The Law" Series is not to be construed as legal advice or opinion.)

*Elizabeth F. Enayati, Esq., Fenwick & West, Two Palo Alto Square, Palo Alto, CA (415-494-0600).* ■

*Continued from page 11*

biotechnology, and medicine by guiding the synthesis of new materials and drugs and predicting their respective effects. A longer-range goal of computer simulations includes the prediction of 3D protein structures from their respective amino acid sequences.

With the introduction of parallel processing techniques into the field of molecular dynamics simulations, these goals become more and more achievable. As indicated in Figure 1, parallel computers are one of the stepping stones on the way to the simulation of real world systems.

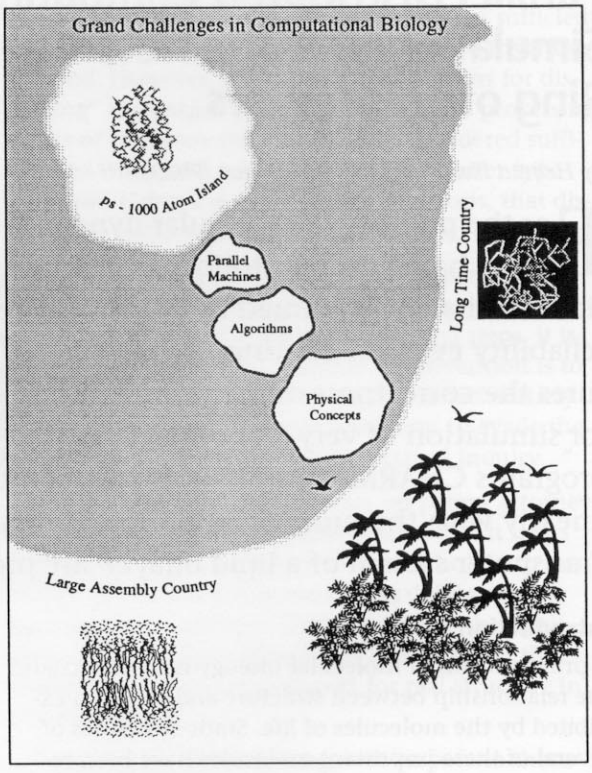


**Figure 1.**
**The grand challenge in computational biology is to simulate real world systems. Currently only systems on the small island of short simulation times (pico seconds) and small system size (1,000s of atoms) can be simulated. With the help of the stepping stones "parallel machines", "faster algorithms", and "new physical concepts" we hope to bridge to the real world. These real world systems consist of 10,000 to 100,000 atoms and have to be simulated over time spans in the range of seconds.**

### What is Molecular Dynamics?

The exact time evolution of a system consisting of many atoms is described by the pertinent quantum-mechanical Schrödinger equations whose complexity—even with only small molecules—renders this description unfeasible for the task of dynamics simulations. Therefore, a classical approximation is used, where the computational demand increases at most like $N_a^2$, where $N_a$ is the number of atoms. In this approximation, atoms are represented as points of mass with a certain charge. The motion of each atom is then determined by summing up all forces acting on it and integrating Newton's equations of motion

$$\left.\begin{aligned} m_i \frac{d^2\mathbf{r}_i(t)}{dt^2} &= \mathrm{F}_i(t) \\ \mathrm{F}_i(t) &= -\nabla_i E(\mathbf{r}_1(t), \ldots, \mathbf{r}_{N_a(t)}) \end{aligned}\right\} \quad (1)$$

Here, $m_i$ and $r_i$ are used to denote the mass and position of the $i$-th atom, respectively. $\nabla_i$ represents $(\frac{\partial}{\partial x_i}, \frac{\partial}{\partial y_i}, \frac{\partial}{\partial z_i})^T$, and $N_a$ is the total number of atoms in the system. $E$ is the empirical energy function, the quality of which is critical for a realistic description of molecules. In the widely used molecular dynamics programs CHARMm[2] and X-PLOR,[4] and in the parallel program EGO,[16] which we have developed for our Transputer system, this energy function is of the form

$$E = \underbrace{E_{bond} + E_{angle} + E_{dihedral} + E_{improper}}_{E_{bonded}} + \underbrace{E_{el} + E_{vdW} + E_{hbond}}_{E_{nonbonded}}, \quad (2)$$

where $E$ is defined as the total energy of the molecule. The individual contributions correspond to the different types of forces acting in the molecule. The first contribution $E_{bond}$ describes the high frequency vibrations along covalent bonds. The second contribution $E_{angle}$ represents the bending vibrations between two adjacent bonds, and the third contribution, $E_{dihedral}$, describes torsional motions around bonds. $E_{improper}$ relates to the motion of one atom relative to a plane defined by three other atoms. All of the previously mentioned energy contributions portray interactions due to a direct chemical bond between atoms. They are often grouped as so-called *bonded* interactions. The remaining three terms describe the *non-bonded* interactions between atoms that are not chemically bonded. $E_{el}$ accounts for the electrostatic (*Coulomb*) energy between the individual atomic charges, $E_{vdW}$ models the *van der Waals* interaction, and $E_{hbond}$ represents the hydrogen bonds.

The method most often used to integrate the Newtonian equations of motion[1] is the Verlet algorithm.[32] The position $\mathrm{r}_i(t + \Delta t)$ of atom $i$ at the instant $t + \Delta t$ is then determined according to the formula

$$\mathbf{r}_i(t + \Delta t) = 2\mathbf{r}_i(t) - \mathbf{r}_i(t - \Delta t) + \mathbf{F}_i(t)\frac{(\Delta t)^2}{m_i} \quad (3)$$

where $\mathrm{F}_i(t)$ as defined in Equation 1 represents the sum of all forces acting on the $i$-th atom at time $t$. The time step $\Delta t$ is determined by the fastest degrees of freedom in the molecule. These are typically bond vibrations of the light hydrogen atoms, and the time step must be no larger than 1.0 $fs$ to insure correct integration of Equation 1. ▶

Because of the many interactions that must be considered, and the small size of the time step, the simulations one would like to carry out are severely hampered by the availability of suitable computer resources. In fact, dynamics calculations up to now have been limited to short simulation periods of a few nanoseconds and to biopolymers of at most a few thousand atoms (see Figure 1). Furthermore, biological macromolecules are usually embedded in an environment of water or lipids, and a realistic description of such molecules should include the surrounding material. The combination of these factors demands the fastest computers available today, which are parallel machines.

*Algorithmic Improvements to Speed-up Computations*

In addition to parallel machines, we need improved and faster algorithms for molecular dynamics simulations if we want to advance on our way towards realistic simulations (see Figure 1).

In the integration of the Verlet equations,[3] most computer time is spent on the evaluation of the non-bonded interactions. These are composed of the electrostatic Coulomb interaction and the van der Waals interaction embodied by a Lennard-Jones potential. For each integration step, due to the long range nature of the Coulomb interaction, $N_a(N_a - 1)/2$ pair interactions must be calculated. The programs CHARMm and X-PLOR avoid the corresponding prohibitive computational effort by introducing a cut-off for these interactions, which for long-range Coulomb forces is not satisfactory. The ensuing error is especially serious for large biopolymers for which inhomogeneous charge distributions are thought to play an important role for stability and function (see Figure 2).

In addition to being of questionable value for the faithful representation of biopolymer dynamics, the introduction of a force cut-off also creates technical problems. The necessity of switching functions to maintain the differentiability of the potential energy function, as well as the need to maintain pair lists to account for the proximity of atoms, entails additional computation and memory requirements which partially offsets the advantage gained through the reduced count of interacting pairs.

Alternative approaches for calculation of non-bonded pair contributions without sacrificing accuracy are the *Fast Multipole Algorithm* developed by Greengard and Rokhlin[11,9,10,12] and a hierarchical distance class approach.[34] The latter method has been implemented in our parallel molecular dynamics program EGO and is described in detail elsewhere.[14,13] The method is based on a spherical subdivision of interatomic distances into several distance classes. The relative motion of atoms that are close in space greatly influences the electrostatic energy, so that these Coulomb force contributions must be calculated at each integration step. Relative motion of atoms separated further apart does not change their Coulomb force contributions as much, and, therefore, these interactions can be evaluated less frequently. The resulting algorithm speeds up computation by a factor of six to ten without a significant loss of accuracy.[14]

## The Transputer Program

As explained in the section on Molecular Dynamics, the computationally most expensive task in molecular dynamics calculations of biopolymers is the evaluation of Coulomb (and van der Waals) forces. Due to the long range character of these forces, each atom interacts with every other atom, leading to a total of $N_a(N_a - 1)/2$ force computations for each integration step. The parallelization strategy for our program and hardware had been adopted to be most suitable to evaluate pair interactions like Coulomb ▶

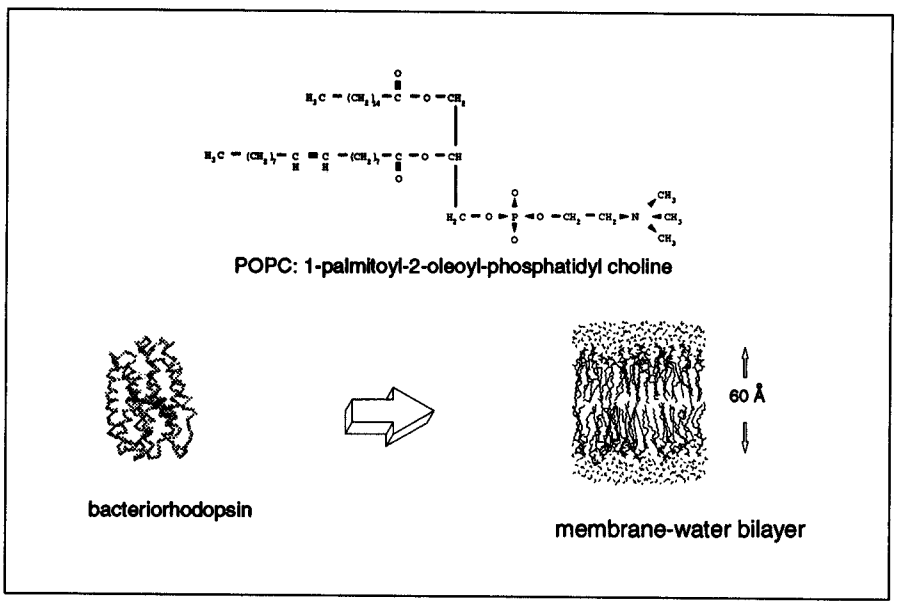


**Figure 2.**
The membrane protein *bacteriorhodopsin*, which has so far only been simulated in vacuum, has been embedded into a membrane-water system. The membrane had been built from 200 POPC lipid molecules.[17] These molecules have charged headgroups with a strong dipole moment. Since the headgroups of the two layers of the membrane are about 60Å apart, a simple force cut-off of the electrostatic interaction at a distance of 10Å to 15Å, as otherwise widely used, would completely neglect that important contribution from the headgroup interaction. Therefore our program EGO uses a special distance class algorithm[14] to incorporate long range electrostatic effects.

*Continued from page 13*

forces. However, all other contributions to the energy function (Equation 2) (and hence the total force) have to be calculated as well and must be integrated in the parallel computational scheme.

*Topology*

In our computational scheme, the atoms of the molecule are assigned to different computational nodes irrespective of the atoms' positions in space or within the molecule (Note 1: Nevertheless it is advantageous to distribute them over the ring according to the order introduced by the protein backbone, as done in EGO by following the sequence introduced by the pdb file) (see Figure 3). All information about these atoms, such as mass, charge, type, and coordinates, is stored in the associated nodes. The evaluation of the forces acting on these atoms also takes place at the Transputer of the node to which the atoms are assigned. In the following discussion, all atoms allocated to a specific Transputer are referred to as the 'own' atoms of that Transputer, all other atoms are the 'external' atoms.

The computation of Coulomb forces is now separated into two terms

$$\mathbf{F}_i = \sum_{\substack{\text{'own'} \\ \text{atoms } j}} \frac{q_i q_j \mathbf{r}_{ij}}{4\pi\varepsilon r^3_{ij}} + \sum_{\substack{\text{'external'} \\ \text{atoms } k}} \frac{q_i q_k \mathbf{r}_{ik}}{4\pi\varepsilon r^3_{ik}} \qquad (4)$$

Here $\mathbf{F}_i$ is the force which acts on atom $i$, $q_i$ is the partial charge of this atom, $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ is the vector joining the atoms at positions $\mathbf{r}_i$ and $\mathbf{r}_j$, and $\varepsilon$ is the dielectric constant.

The first term in equation 4 can be evaluated by all nodes in parallel, since it involves only information of 'own' atoms $i$ and $j$, which is always known locally at each node. The second term in equation 4, however, involves 'own' atoms $i$ as well as 'external' atoms $k$ and therefore requires some communication.

The systolic double ring topology,[18] which is depicted in Figure 4, provides the communication channels needed. The network Transputers (T800), one of which can be entirely devoted to the calculation of hydrogen bond[1] (h-bond node, shown in light grey in Figure 4), and a host Transputer, are joined by a bidirectional Transputer link (two antiparallel, unidirectional occam II channels). This forms the first ring. A second ring is established using the two remaining links on each network Transputer. It is neither desirable nor necessary (and without a fifth link not possible) to include the host Transputer in this second ring. There is also no need for the h-bond node to be tied into this second ring.

Using the first ring, each Transputer can send a copy of the information it has about its 'own' atoms (coordinates) to its neighbor in clockwise direction (on the dark black channel in Figure 4, page 15). After the coordinates have been received, each Transputer can calculate that part of the sum in Equation 4 concerning the 'external' atoms it just received from its neighbor. In the next step, the information about the 'external' atoms is passed to the next neighbor in clockwise direction by each node. Knowing a new set of 'external' atoms, each node can calculate another part of the 'external' sum in Equation 4. This process is repeated until all pair interactions have been computed. By then, each node will have received its 'own' atoms back, which have circulated one full revolution on the ring. Using the total Coulomb force component, along with the bonded force contributions of its 'own' atoms, the entire force $\mathbf{F}_i$ acting on each atom is known, and all nodes can integrate one step according to Equation 3.

By applying Newton's law $\mathbf{F}_{ki} = -\mathbf{F}_{ki}$ it is possible to cut the computation time to about one half, avoiding redundant calculations. This is achieved by sending partial force arrays $\mathbf{F}_{ki}$ along with the coordinates ▶

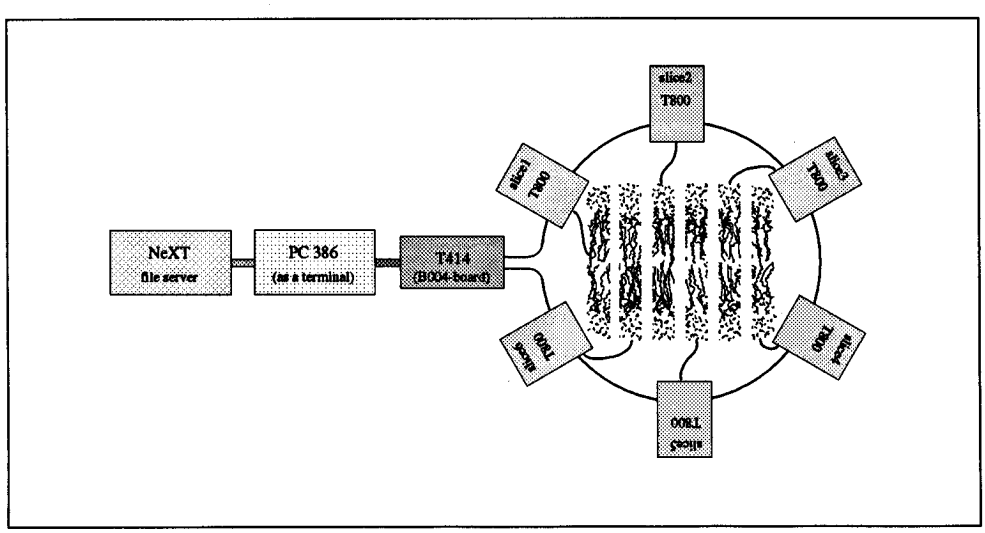


**Figure 3.**
**Schematic diagram of the systolic double ring network used in our application. The system, here the membrane bilayer covered with water, is sliced in as many pieces as there are computational nodes in the ring. Then the atoms in each slice are assigned to a node, illustrated by the little curved lines, as the 'own' atoms of that node.**

on the force.array channel (medium grey channel in Figure 4.

One drawback of this method is that the strategy can effectively only use an odd number of computational nodes. Since we have six nodes on each board, i.e. always an even number of nodes available, one node would remain idle through-out the simulation.[16,15] However, making use of the multiple instructions, multiple data (MIMD) structure of the Transputer network, this node can be devoted to the calculation of hydrogen bonds[1] without significantly slowing down the other calculations.

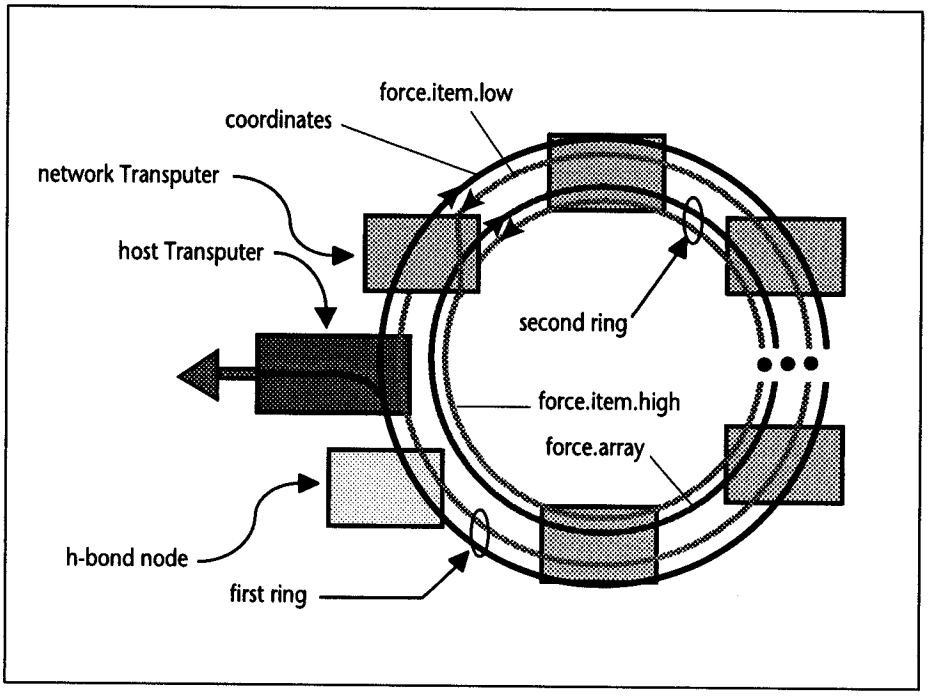


**Figure 4. Communication channels for the double ring topology used in our parallel computer. See text for details.**

*Deadlock-free Force Communication*

Two kinds of forces are produced: force items and force arrays. All the pair interactions can be evaluated immediately after coordinates of external atoms have arrived at a node, as they involve only one external atom. The forces produced during those computations can be sent as arrays, which is much more efficient than sending them as single force items for each atom. For this purpose we use the force.array channel channel depicted in medium grey in Figure 4. Using a separate channel for force arrays and for coordinate arrays speeds up communication since the channels reside on different links that can be operated in parallel. (Note 2: Technically, the force arrays are out of phase with the coordinates by exactly one transfer step. This reflects that the coordinates have to be known *prior* to the computation, while the forces are known only *after* the computation.) Different systolic loop methods are described elsewhere.[27]

The other type of forces stem from interactions between three (angle interactions) and four (dihedral and improper interactions) particles. For their computation, coordinates from up to four different nodes need to be available. For this purpose the coordinates of atoms involved in many-body interactions are saved while they are passed around in the ring. When all the necessary information to compute a many body interaction is available to a node, the computation is carried out and the forces are sent one by one as force items to the node that owns the respective atom, using the force item channels (light grey in Figure 4).

All new force items are multiplexed into the force.item.low channel. Each node listens on this channel for items targeted to it. This raises a deadlock problem, since a situation is possible where all the nodes might want to forward a force item but can not do so because there is already an item stuck in the channel. To avoid this problem a second force item ring, force.item.high, is introduced. No node is allowed to multiplex items into this ring. The only allowed operations are forwarding of items and removal of items addressed to this very node.

However, there is exactly *one* node that passes information from the force.item.low channel to the force.item.high channel (as shown in Figure 4); thus the rings become a spiral. The force.item.high ring can never deadlock since the only allowed operations are removal and forwarding for force items. This implies that there is always one node on the force.item.low channel that cannot deadlock (namely the one that crosses both rings). This guarantees the deadlock freedom of the whole system.[5]

*Compatibility with Other Programs*

Compatibility with standard molecular dynamics packages was a key issue in the development of our program. The input files used (containing the atomic coordinates, the structure information, and force field parameters) can easily be obtained from the file format used by X-PLOR. The output files containing the trajectories of the individual atoms can also be converted into X-PLOR format (Note 3: This feature was implemented by M. Tesch.) for detailed analysis with X-PLOR. ▶

*Continued from page 15*

*Dissemination*

Besides the installation at the UIUC, several other universities and research institutions installed EGO on their systems (see Figure 5). Recently M. Schaefer and H. Heller ported the approximately 20,000 lines of occam II code to the C-language to make EGO available also on parallel non Transputer-based machines, such as the CM5 of Thinking Machines Corporation (TMC), the Paragon/XP of Intel, or networks of workstations. The latter effort builds on the coordination language CHARM[22,7,29] which was developed by Prof. Kale at the UIUC. This will enable many researchers who do not have access to a Transputer system to use EGO for their research.

To try out the EGO program and/or a Transputer system, the NIH Resource "Concurrent Biological Computing" has made available a Transputer guest account on one of its machines. This guest account allows access to a 6 Transputer network which allows for the simulation of small molecules (up to 1,500 atoms) using the program EGO. Further details on how to use this service are provided in Appendix A.

## Hardware Development

In November 1987, when we began our efforts to develop molecular dynamics algorithms for Transputers, to our knowledge no commercial hardware meeting our specifications (large, parity checked DRAM on each node, status indicator, compact design) was available. Therefore, we designed and built our own boards.

The hardware consists of a systolic double ring[25] of Transputer nodes (see Figure 3), and is fully described elsewhere.[13,16,15] The main parts of each computational node are an INMOS T800 Transputer running on 22.5 MHz, 4 MBytes of parity checked DRAM (100 ns access time), a parity checking logic and a status indicator. The latter signals the internal state of each node to the user by means of three color-coded LED's (one for a set error flag, one for a parity error, and one for access of the external DRAM). Deadlock situations and an uneven distribution of the workload can be easily detected by observing the DRAM access indicator lights.

Six computational nodes fit onto one six-layer, double eurocard sized (160 mm by 233 mm) board. Ten of these boards fit into one 19" chassis, totalling 60 nodes which for molecular dynamics calculations match the computational power of a Cray 2 processor (see the section on Benchmarks). Such a powerful machine can be placed on a desktop and does not take up more space than a PC (see Figure 6, page 17).

All link connections determining the topology of the network (here a systolic double ring) are provided by the backplane of the chassis, which can be exchanged to allow for different topologies.

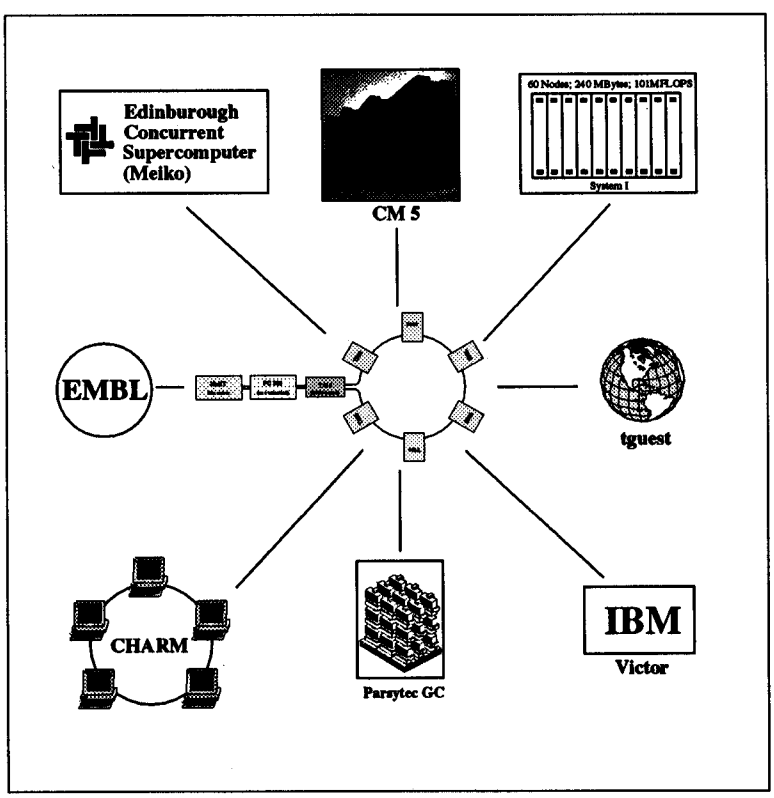


Figure 5. Dissemination of the parallel molecular dynamics program EGO. Most installations use the occam II version, but the CM5 and CHARM[22] installations use the new C-language version of EGO.

Parity checking is important, since we run a system with 60 nodes and 240 MBytes of solid state memory. With such a large system, a parity error is likely to occur every other month. Assuming a runtime of several months for the molecular dynamics code, this probability is unacceptably high. Together with an automatic restart mechanism in the software, the parity checking allows sufficiently long, unsupervised dynamics calculations without the risk of corrupting data due to memory faults.

In addition to occam II and the TDS3 of INMOS, the C compilers Par.C by Parsec and ANSI-C by INMOS are available. Although designed for molecular dynamics simulations, we successfully run neural network simulations, magnetic resonance imaging (MRI) simulations, and percolation simulations on our Transputer systems. Using the PC-SERVER of Luigi Rizzo, not only the UNIX based systems but also the IBM-PC based systems are fully accessible ▶

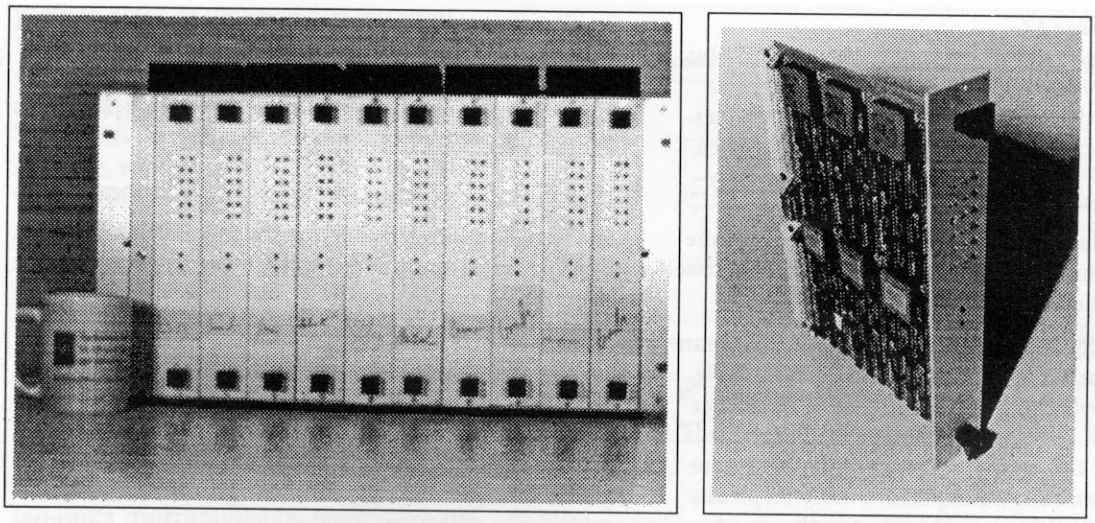

Figure 6.
Left: A 60 node/240 Mbyte DRAM Transputer-based parallel computer with the performance of a Cray 2 processor. This machine was designed and built for $60,000 by members of our group. Nine graduate students and a postdoctorate each soldered a board (note their signatures on the respective boards). The machine, programmed for large-scale molecular dynamics simulations, is shown running a simulation of a 24,000 atom water-membrane system using the X-PLOR compatible program EGO; the 60 lit status lights indicate that all nodes run at full speed demonstrating excellent load balance. The cup on the left indicates the small size of the machine. Right: Picture of a single board. Dominant component of the footprints of the six nodes is the Transputer chip; the 4 MBytes of parity checked memory for each node are mounted on the other side of the board.

remotely over the network from virtually everywhere in the world.

As shown in Figure 3 and Figure 7 (page 18),the systolic double ring is interfaced to a PC (or a Silicon Graphics Workstation). Using Sun's Network File System (NFS) large disks with hundreds of megabytes are connected to the Transputer system to store the huge amount of output data, typically 70 Mbytes per day. Since the output is split over several files it is even possible to look at a trajectory and analyze it (using X-PLOR on a conventional machine) *while* later portions of it are still being computed by EGO. Such preliminary analysis allows detection of obvious errors and monitoring of the progress of the computation.

## Benchmarks

*Comparison of the Transputer System with Other Systems*

Several benchmarks comparing the molecular dynamics performance of our Transputer system with other computers have been carried out. Different programs were used on the individual machines; the results are given in Table 1 (page 19). To compare the performance of our Transputer-based parallel computer with other machines, we computed on several machines a short (100 steps) molecular dynamics trajectory of *bacteriorhodopsin* with 2,166 atoms, *bacteriorhodopsin* with additional water totalling 4,039 atoms, and the membrane bilayer with water cups and embedded *bacteriorhodopsin* totalling 23,208 atoms.

The distance class algorithm[14] used by EGO provides a speed-up of six to ten (compared to calculations without cut-off), but does not introduce the errors connected with neglecting far range Coulombic and van der Waals interactions. Therefore, it must be considered as being somewhat in between those methods using a cut-off and those computing all $N_a(N_a - 1)/2$ interactions, as indicated in Table 1.

From Table 1, one can easily see that a 60~node Transputer machine running EGO outperforms most of the other tested supercomputers while it costs only a fraction of their price. Our actual costs were about $60,000 for the whole machine, which excludes costs for development and assembly.

A detailed performance analysis of EGO can be found elsewhere.[30]

*Scaling Performance of the Transputer System*

Aside from the absolute performance, another key property of parallel machines is how the performance scales with the number of nodes working on a given problem. This scaling performance is shown in Figure 8 (page 19).

The performance is given by the number of integration steps that can be completed per time unit, i.e. per second. For an ideal machine this performance would increase linearly with the number of nodes; the actual behavior comes very close to this linear dependence. ▶

*Continued from page 17*

The deviation from the linear behaviour is smaller for the larger system. Experiments showed that a minimum number of ~50 atoms per node is necessary to achieve a good scaling of the performance.

### Simulation Results

(Note 4: A detailed description of this membrane simulation is currently being published with Michael Schaefer elsewhere).

Figure 9 (page 20) shows some results of the membrane simulation. The deuterium order parameter profile, at the top, shows the typical decrease of the order parameter with increasing chain position (towards the free end of the chain). The absolute values are higher than those found experimentally (typically between 0.1 and 0.2), probably due to the short simulation time.

The water density in the membrane region, shown in the center of Figure 9, was computed at three different times. The increased penetration of water into the lipid region at later times is clearly visible.

The picture at the bottom in Figure 9 shows the trajectories of the centers of mass of each lipid molecule as a projection into a plane perpendicular to the membrane plane. The inward pointing trace of the outer (restrained) lipid molecules is a result of the contraction of the reference coordinate set. Due to these restraints the diffusion of the outer lipids is remarkably smaller that that of the free lipids in the center. The lipids show considerable movement perpendicular to the membrane plane, but the simulation time of 115 ps is too short to show lateral diffusion with lipids exchanging places.

### The Next Generation of Transputers: T9000

In October 1985, INMOS introduced the Transputer family.[8] Since then, several improved members have been added to the family. In 1987, the T800 was introduced,[6] and with only a few modifications it has remained a top-of-the-line product. We currently employ the T800 processor in our parallel computers.

While other manufacturers introduced their new processors (e.g., the i860 by Intel) over the past four years, INMOS worked on the design of the T9000 which was officially unveiled in April 1991. Unfortunately INMOS was not able to deliver actual products at that time and as of this past May, more than a year has passed without the T9000 being available. The new announcement of INMOS is that T9000 will be available by the end of 1992.

The T9000 provides an increase in speed of about a factor of ten in comparison to its predecessor, the T800, in both communication as well as computation.[21] This improvement can easily be utilized as the new T9000 is binary compatible with the T800. It is possible to carry over all the programs developed for the T800 Transputer, with only minor modifications necessary to handle the improved link structure that allows transparent communication between any two T9000s.

From our experience with current Transputers (T800), it is very reasonable to expect that the T9000 will achieve the claimed values for sustained performance, which are 60 MIPS (150 MIPS [Note 5: Mega Instructions Per Second] peak) and 10 MFLOPS [Note 6: Mega FLoating point Operations Per Second] (20 MFLOPS peak)[21] in our molecular dynamics simulation application using our program EGO.

The T800 (see Figure 10, page 21, left side) is a 32 bit processor with a 64 bit floating point coprocessor and four independent communications agents (links) integrated on a single chip.[19][page 5] The links ▶

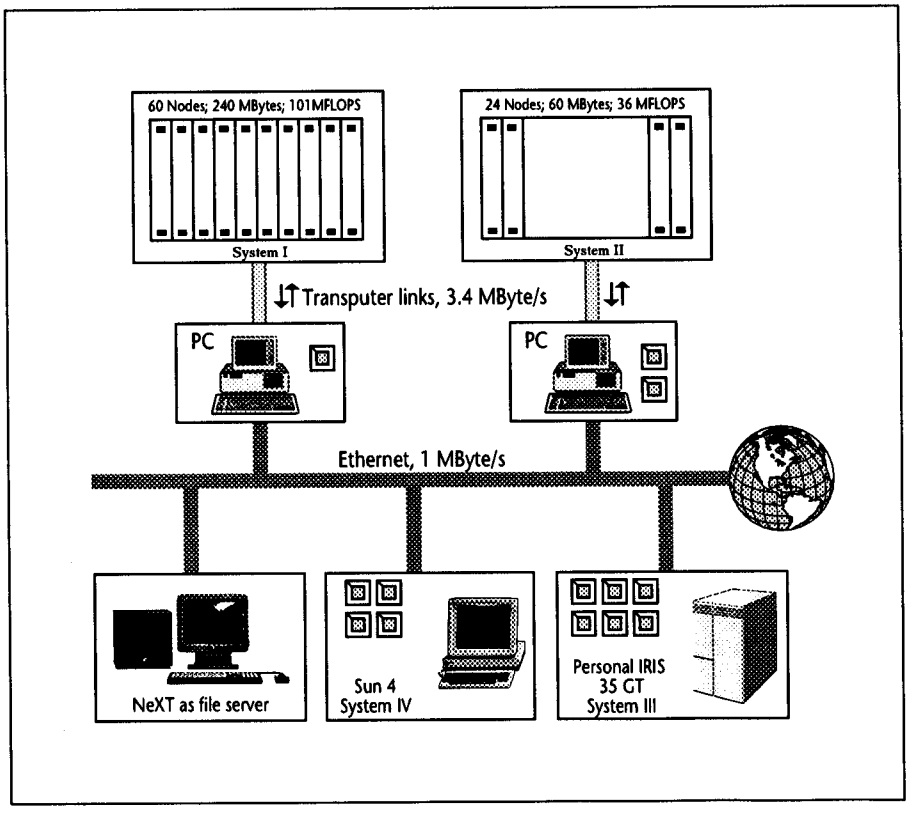


**Figure 7. Networking of the Transputer systems in the theoretical biophysics group of K. Schulten.**

provide point to point communication between any two Transputers. Having only four links, each Transputer can connect with only four other Transputers. This limits, in general, the portability of software, which must be adapted to different network topologies. Also, the programmer must address the problem of communication explicitly; for instance, routers must be written and included in the source text.

The IMS C004, a 32 x 32 programmable crossbar switch[20] was designed to increase the connectivity in a network of T800 Transputers. It can connect any of its 32 input links with any of its 32 output links, programmed by commands on a control link. Unfortunately, it introduces a 1.75 bit delay in the exchanged messages (reducing the bandwidth from 1.7 to 1.3 MBytes/ per link), and has a very high latency for reconfiguration. Therefore it is not suitable for *dynamic* reconfiguration of Transputer networks.

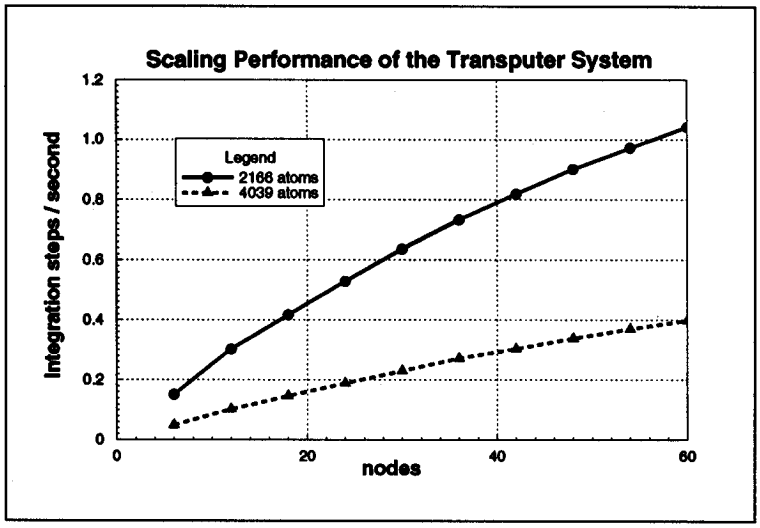


Figure 8. Scaling of the performance with the number of nodes. Since only complete boards with six nodes could be removed, the measured values (dots and triangles) are in increments of six nodes. The scaling behavior of the performance was measured for two systems with 2,166 atoms and 4,039 atoms, respectively.

The communication problems of the T800 and the C004 are remedied by the virtual channel processor of the T9000 (see Figure 10, right side) in conjunction with the new C104 crossbar switch with submicrosecond latency. The virtual channel processor can multiplex any number of links onto a single physical link, making them "virtual links".[23] Each message on a channel is split into a sequence of packets; packets from different messages on different virtual links can be interleaved over each physical link, thus preventing deadlock. The C104 will be able to route *individual messages* to their destinations, rather than simply providing a reconfigurable link topology like the C004. A switching latency of about $\mu$s (low load conditions) and a total bandwidth of about 5 GBytes/s are expected. The following explanation is taken from Reference 26:

> "Inmos chose interval routing, which is complete (every packet eventually arrives), deadlock-free, inexpensive, fast, scalable (good for any size network), versatile (good for any type topology), and can be near optimal (shortest routing followed). A number assigned to each ▶

**Table 1.**

| Program | Machine | 2,166 atoms cut-off | 2,166 atoms no cut | 4,039 atoms cuto-off | 4,039 atoms no cut | 23,208 atoms |
|---|---|---|---|---|---|---|
| EGO | GC: 128 T805 nodes | 0.66 | | 1.33 | | 32.1 |
| EGO | 60 T800 nodes | 0.97 | | 2.51 | | 83.8 |
| EGO | CM-5 32 nodes | ~58 | | – | | – |
| EGO | CM-5 64 nodes | ~90 | | – | | – |
| MD | CM-2, 32k procs | n/a | – | n/a | 2 | – |
| X-PLOR1.5 | CRAY 2 | 0.64 | 3.39 | 3.52 | 20.4 | – |
| X-PLOR1.5 | Convex C2 | 3.3 | 21.3 | 14.0 | 88.55 | – |
| X-PLOR1.5 | Stardent 3000 | 8.2 | 46.5 | 28.5 | 196.7 | – |
| X-PLOR1.5 | SGI 4D/220GTX | 15.54 | 121.8 | 43.88 | – | – |
| X-PLOR1.5 | Stardent 1500 | 18.22 | 123.16 | 62.2 | – | – |

The table shows the average CPU time taken for one integration step of three sample systems with 2166 atoms, 4039 atoms, and 23,208 atoms respectively. The machine entry "GC" stands for the Parsytec GCel machine, which we could use for a couple of days.The timings for the CM-5 are very preliminary as that version of EGO is not yet completely debugged and not at all optimized. The implementation effort is still ongoing. If not noted differently, the program is run using only one CPU. The two timings for X-PLOR indicate computations done with cut-off (11 Å) and without cut-off. MD[33] does not allow for a cut-off. EGO, the dynamics program on the Transputer machine, uses a distance class algorithm.[14]

*Continued from page 19*

node is used for its address. At every routing switch each link knows which header values it should recognize. Intervals are nonoverlapping and numbered so every header falls into just one interval. Each interval is a pointer to the subtree containing all these processor numbers. The attraction in this scheme is that routing can be computed with a single comparison. The C104 allows 1 or 2 Byte headers, giving up to 65,536 Transputers in a single network.

"Algorithms exist to prevent cyclic paths that are deadlock free. An optimal labeling scheme for an arbitrary network is unknown, but the more common topologies such as rings, trees, and two-dimensional arrays are known. Grid and hypercube schemes can be constructed from these topologies.

"When too many messages go through a single node, a hot spot develops, stalling a large number of packets. This can be avoided by evenly distributing the flow. The C104 can perform this using universal routing by choosing a random node to initially send the packet to, where it is forwarded to its destination. Latency is increased and throughput is decreased, but the worst-case performance is near optimum...

"Packet switching removes certain restrictions in occam II. The present version is a static language in which resources must be determined at compile time, so new processes cannot be added at run time on remote processors. Also each channel must be explicitly allocated to a physical link at compile time. The T9000 and the C104 allow full occam II to be implemented."

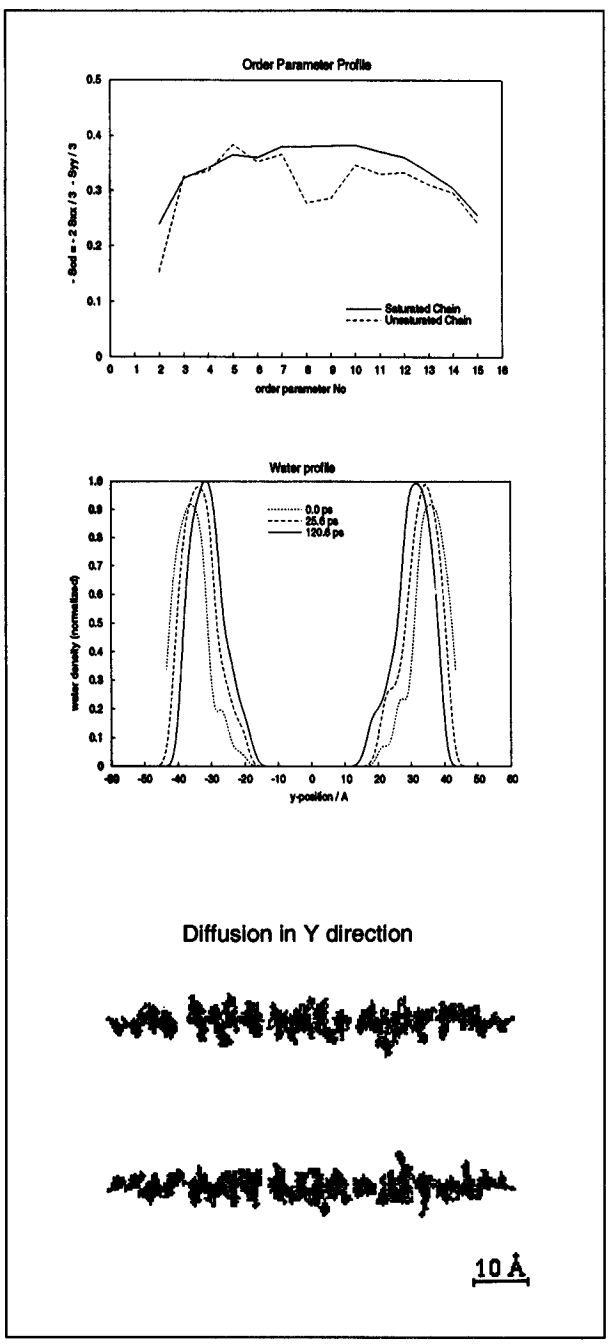


**Figure 9.** Simulation results from the equilibration phase of the membrane-water system (without *bacteriorhodopsin*). Top: Order parameter profile for the two fatty acid chains of the lipid molecule. The decrease in the order parameter for the carbon atoms at the end of the chain (high order parameter No.) is typical. Center: The water penetration profile, recorded at the start, at an intermediate point, and at the end of the simulation, shows that the penetration of water molecules into the headgroup region increases with time. Bottom: Shown is the Y-component of the centers of mass of each lipid molecule over the course of the simulation.

Portability to any routed array of Transputers is possible without any change, although efficiency will be determined by the topology of the underlying hardware. We expect an increase in speed by at least a factor of ten between a T800 system and a T9000 of equal size. The improved communications scheme eliminates the need for our software routers, making the EGO program shorter, less complicated, and faster.

*Helmut Heller and Klaus Schulten, Ph.D., Beckman Institute and Department of Physics, University of Illinois, 405 N. Mathews Avenue, Urbana, IL 61801.*

**Acknowledgements:**

We would like to thank Lyndon Clarke from the University of Edinburgh who contributed the idea of the *double* ring topology and who helped to implement it on the Edinburgh Concurrent Supercomputer (ECS).

We would further like to thank Michael Schaefer who helped a lot with the membrane simulations and Helmut Grubmüller who was one of the chief designers of the whole Transputer system.

This work was supported in part by the Beckman Institute at the University of Illinois in Urbana-Champaign and the National Institutes of Health Resource "Concurrent Biological Computing" under grant number 1 P41 RR05969*01.

## Appendix A.
## How to use the Transputer Guest Account

Contact: Helmut Heller

Phone: (217) 244-6914

E-mail: heller@lisboa.ks.uiuc.edu

Office: Beckman 3rd floor, room #3143 ▶

Address: University of Illinois
Beckman Institute
Attn: Mr. Helmut Heller
405 North Mathews Avenue
Urbana, Illinois 61801, U.S.A.

logon: tguest@oslo.ks.uiuc.-edu, IP-number: 128.174. 214.4. You will obtain the password if you send a short e-mail stating your name, address, and why you would want to use the guest account to heller-@lisboa.ks.uiuc.edu

## Appendix B. What is available

### *Hardware*

- Production sites
  - 60 T800 nodes (22.5 MHz) with 4 MByte each and parity checking,
  - host: B004 (T414 (17.5 MHz) with 2 MByte with parity checking) in '386 PC
  - 26 T800 nodes (20 MHz, 22.5 MHz) with 1 MByte
  - host: B008 (T800 (25 MHz) with 2 MBytes) in AT compatible PC
- Development sites
  - 5 T800 nodes (20 MHz) with 1 MByte each
  - host: B014 (T800 (25 MHz) with 4 MByte) in Silicon Graphics IRIS4D machine, supports remote login (main tguest machine)
  - 4 individual T800 nodes with 1 MByte each on a Parsytec VMTM board supporting four independent users hosted in a SUN4

### *Software*

- Transputer Development System from INMOS, TDS3 D700E
  - occam II compiler producing very fast code
  - folding editor
  - network debugger
  - very stable environment
- Par.C from Parsec, V 1.3
  - C compiler for parallel programs
  - the executable code produced seems to be slower than that of the occam II compiler
- Helios from Parahelion, V 1.1
  - UNIX-like operating system for transputer networks up to 20 Transputers
  - C compiler for sequential programs
  - V 1.0 proved to be highly unstable (therefore not used by us)

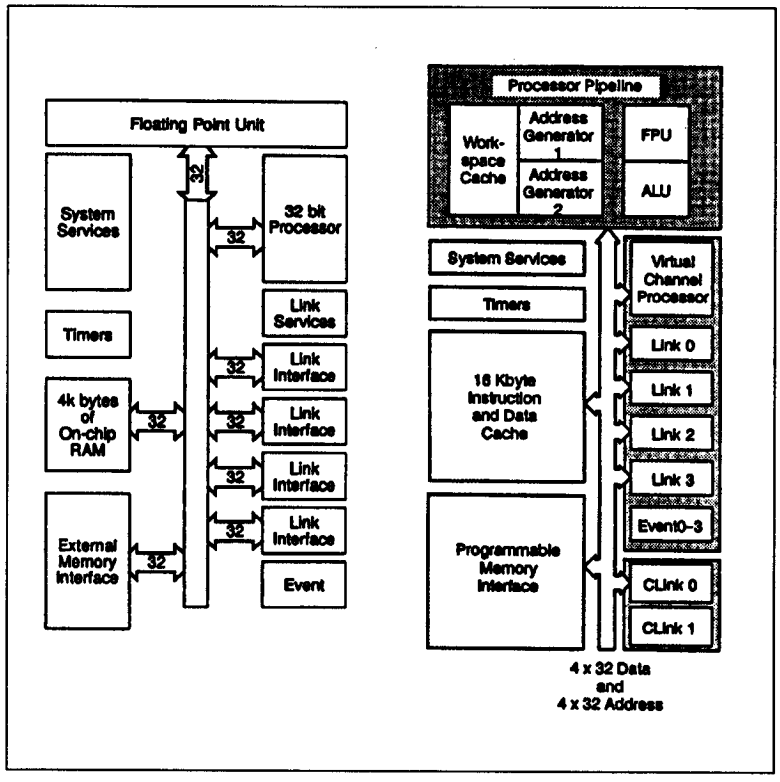


**Figure 10.** Comparison of the T800 and the T9000 Left: Block diagram of the T800.[20] Integrated on one chip one finds not only ALU, system services and a programmable memory interface, but also a 64-bit FPU, 4 KBytes of on-chip RAM, and four independent link interfaces. Right: Block diagram of the T9000.[21] All the basic building blocks of the T800 are visible: ALU, system services and a programmable memory interface, 64-bit FPU. New is an additional workspace cache. The cache/RAM has been extended to 16 KBytes, and, most importantly, the four link interfaces have been equipped with a virtual channel processor.

### *Parallel Applications*

- EGO, a molecular dynamics simulation program, I/O-compatible to X-PLOR format
- software for simulation of Magnetic Resonance Imaging (written in C utilizing a farm concept, B. Puetz)
- software for simulation of neuronal networks (K. Obermayer)
- software for investigations of percolation phenomena (C. Kurrer)

### References

1. Boehncke, K.; Heller, H.; Grubmüller, H. and Schulten, K. (1990) "Molecular Dynamics Simulations on a Systolic Ring of Transputers," in A. S. Wagner, ed., *Naturg 3: Transputer Research and Appplications 3,"* 83-94, North American Transputer Users Group, IOS Press, Amsterdam.
2. Brooks, B. R.; Bruccoleri, R. E.; Olafson, B. D.; States, D. J.; Swaminathan, S. and Karplus, M. (1983) "CHARMM: A Program for Macromolecular Energy, Minimization, and Dynamics, Structure, and Thermodynamics," *J. Comp. Chem.* 4(2), 187-217.
3. Brooks, C. L. III; Karplus, M. and Pettitt, B. M. (1988) *Proteins: A Theoretical Perspective of Dynamics, Structure, and Thermodynamics,* John Wiley & Sons, New York. ▶

*Continued from page 21*


4. Brünger, A. T. (May, 1988) *X-PLOR,* The Howard Hughes Medical Institute and Department of Molecular Biophysics and Biochemistry, Yale University, New Haven, CT.
5. Dally, W. J. and Seitz, C. L. (1986) "The Torus Routing Chip," *Distributed Computing* 1, 187-196.
6. Eckelmann, P. (1987) "Transputer der 2. Generation," *Elektronik,* Issues 1, 2, and 3.
7. Fenton, W.; Ramkumar, B; Saletore, V.; Sinha, A. and Kale, L. V. (in press) *Supporting Machine Independent Programming on Diverse Parallel Architectures.*
8. Ghee, S. (February, 1987) "IMS B004; IBM PC add-in board," *INMOS,* Technical Note 11, #72TCH01100.
9. Greengard, L. (1988) *The Rapid Evaluation of Potential Fields in Particle Systems,* MIT Press, Cambridge.
10. Greengard, L. and Gropp, W. D. (1988) "A Parallel Version of the Fast Multipole Method," Technical Report, Rept. YALEU/DCS/RR-640.
11. Greengard, L. and Rohklin, V. (1987) "A Fast Algorithm for Particle Simulation," *J. Comp. Chem.* 73, 325-348.
12. Greengard, L. and Rokhlin, V. (1988) "On the Efficient Implementation of the Fast Multipole Algorithm," Technical Report, Rept. YALEU/DCS/RR-602.
13. Grubmüller, H. (1989) *"Dynamiksimulation sehr großer Makromoleküle auf einem Parallelrechner,"* Master's Thesis, Physics Dept., Technical University of Munich, Germany.
14. Grubmüller, H.; Heller, H.; Windemuth, A. and Schulten, K. (1991) "Generalized Verlet Algorithm for Efficient Molecular Dynamics Simulations with Long-Rnage Interactions," *Molecular Simulation* 6(1-3), 121-142.
15. Heller, H. (1988) *Dynamiksimulation sehr großer Makromoleküle am Beispiel des photosynthetischen Reaktionszentrums von Rhodopseudomonas viridis,"* Diploma Thesis, Physics Department, T 30, Technical University of Munich, James-Franck-Street. D-8046 Garching, Munich Germany.
16. Heller, H; Grubmüller, H. and Schulten, K. (1990) "Molecular Simulation on a Parallel Computer," *Molecular Simulation* 5, 133-165.
17. Heller, H.; Schaefer, M. and Schulten, K. (1992, in prepartion) "Construction, Molecular Dynamics Simulation, and Analysis of a Lipid Bilayer," *Biochemistry.*
18. Hillis, W. D. and Barnes, J. (1987) "Programming a Highly Parallel Computer,*Nature* 326, 27.
19. (July 1987) INMOS: "Reference Manual: Transputer Architecture," #72TRN04803.
20. (1989) INMOS: "The Transputer Databook."
21. (1990) INMOS: "H1 Transputer," Advance Information.
22. Kale, L. V. (1990) "The Chare Kernel Parallel Programming Language and System," *Proc. of the International Conf. on Parallel Processing* 2, 17-25.
23. May, D. and Thompson, P. (April, 1990) "Transputers and Routers: Components for Concurrent Machines," Technical Report, INMOS.
24. McCammon, J. A. and Harvey, S. C. (1987) *Dynamics of Proteins and Nucleic Acids,* Cambridge University Press, Cambridge.
25. Ostlund, N. S. and Whiteside, R. A. (1985) "A Machine Architecture for Molecular Dynamics: The Systolic Loop," in B. Venkataraghavan and R. J. Feldmann, eds., "Macromolecular Structure and Specificity: Computer-Assisted Modeling and Applications", 195-208, Annals of the New York Academy of Sciences **439,** New York.
26. Puntain, D. (1990) "Virtual Channels: The Next Generation of Transputers," *Byte,* E & W, 3-12.
27. Raine, A. R. C.; Fincham, D. and Smith, W. "Systolic Loop Methods for Molecular Dynamics Simulation Using Multiple Transputers," *Computer Physics Communications* **55,** 15-30.
28. Schwyzer, R. (1991) "Peptide-Membrane Interactions and a New Principle in Quantitative Structure-Activity Relationships," *Biopolymers* **31,** 785-792.
29. Shu, W. W. and Kale, L. V. (November, 1989) "A Dynamic Load Balancing Strategy for the Chare Kernel System," *Proceedings of Supercomputing '89,* 389-398.
30. Sinha, A.; Heller, H. and Schulten, K. (in preparation) "Performance Evaluation of the Parallel Molecular Dynamics Program EGO," *Computer Physics Communications.*
31. van Gunsteren, W. F. and Berendsen, H. J. C. (1990) "Moleküldynamik-Computersimulationen; Methodik, Anwendungen und Perspektiven in der Chemie," *Angew. Chemie* **102,** 1020-1055.
32. Verlet, L. (1967) "Computer 'xperiements' on Classical Fluids. I. Thermodynamical Properties on Lennary-Jones Molecules," *Physical Review* **159,** 98-103.
33. Windemuth, A. and Schulten, K. (1991) "Molecular Dynamics on the Connection Machine," *Molecular Simulation* **5,** 353-361.
34. Windemuth, A. (August, 1988) *Dynamiksimulation von Makromolekülen,* Master's Theis, Physics Department, Technical University of Munich, T 30, James-Franck-Street, D-8046 Garching, Munich, Germany. ■